A Learning and Masking Approach to Secure Learning

Linh Nguyen, Sky Wang, and Arunesh Sinha

University of Michigan, Ann Arbor, USA
{lvngyuen,skywang,arunesh}@umich.edu

Abstract. Deep Neural Networks (DNNs) have been shown to be vulnerable against adversarial examples, which are data points cleverly constructed to fool the classifier. In this paper, we introduce a new perspective on the problem. We do so by first defining robustness of a classifier to adversarial exploitation. Further, we categorize attacks in literature into high and low perturbation attacks. Next, we show that the defense problem can be posed as a learning problem itself and find that this approach effective against high perturbation attacks. For low perturbation attacks, we present a classifier boundary masking method that uses noise to randomly shift the classifier boundary at runtime. We also show that both our learning and masking based defense can work simultaneously to protect against multiple attacks. We demonstrate the efficacy of our techniques by experimenting with the MNIST and CIFAR-10 datasets.

1 Introduction

Recent advances in deep learning have led to its wide adoption in various challenging tasks such as image classification. However, the current state of the art has been shown to be vulnerable to *adversarial examples*, small perturbations of the original inputs, often indistinguishable to a human, but carefully crafted to misguide the learning models into producing incorrect outputs. Recent results have shown that generating these adversarial examples are inexpensive [9]. Prior work has yielded a lot of attack methods that generate adversarial examples, and defense techniques that improve the accuracy on these examples (see related work). However, attacks and defenses have followed the cat-and-mouse game that is typical of many security settings. Further, traditional machine learning theory assumes a fixed stochastic environment hence accuracy in the traditional sense is not a meaningful measure of performance in the presence of an adversary.

In this paper, we pursue an approach informed by our *first contribution*: a definition of *robustness of classifiers* in the presence of an adversary. Towards the definition, we define an exploitable space by the adversary which includes data points already mis-classified (errors) by any given classifier and any data points that can be perturbed by the adversary to force mis-classifications. Robustness is defined as the probability measure of the exploitable space. We also analyze why accuracy fails to measure robustness. Using our formal set-up we categorize

known attacks into high and low perturbation attacks, and explain why defenses against one type of attacks does not work against the other type.

Our *second contribution* is a defense technique: *defense learning neural network* (DLN) against high perturbation attacks. A DLN D is a DNN that, given any classifier C attacked by an attack technique A, takes in an adversarial example $A(x)$ and aims to generate benign example $D(A(x))$ that *does not* lie in the mis-classified space of C. For non-adversarial inputs the DLN is encouraged to reproduce the input as well as make the classifier predict correctly. The DLN is *prepended* to the classifier C acting as a sanitizer for C. We show that DLN allows for attack and defense to be set up as a repeated competition leading to more robust classifiers. While DLN works efficiently for attacks that produces adversarial examples with high perturbation, such as fast gradient sign method [9] (FGSM), it is not practical for low perturbation attacks (illustrated in Fig. 3) such as Carlini-Wagner [5] (CW).

Our *third contribution* is a defense against low perturbation attacks that we call *noise augmented classifier* (NAC) which randomly shifts the classifier separator by injecting a very small noise at the last layer of the DNN classifier during runtime. The small noise *randomly* shifts the separator on each invocation, but not by much, thereby ensuring original accuracy is maintained, yet also fools low perturbation attacks. NAC alone defends against the low perturbation CW attack, but as expected fails against high perturbation FGSM attack.

Finally, we show that DLN and NAC can work together, thereby enabling simultaneous defense against both high and low perturbation attacks. We tested our approach on two datasets: MNIST and CIFAR-10, and the resultant classifier was robust to both FGSM and CW. All missing proofs are in the full version.

2 Model and Approach

Attack Model: First, we use *inference phase* of a classifier to mean the stage when the classifier is actually deployed as an application (after all training and testing is done). The attacker attacks *only* in the inference phase and can channel his attack *only* through the inputs. In particular, the attacker cannot change the classifier weights or inject any noise in the hidden layers or access any internal values when the DNN predicts in the inference phase. The attacker has access to the classifier weights, so that it can compute gradients, if required. The attacker's goal is to produce adversarial data points that get mis-classified by the classifier, and are not a garbage noisy image.

Notation: Let the function $C : X \to Y$ denote a classifier that takes input data points with feature values in X and outputs a label among the possible k labels $Y = \{1, \ldots, k\}$. Further, for DNNs we define $C_p : X \to \Delta Y$ as the function that takes in data and produces a probability distribution over labels. Thus, $C = \max\{C_p(x)\}$, where C is the maximum component of the vector $C_p(x)$. Let $H(p, q)$ denote the cross entropy $-\sum_i p_i \log(q_i)$. For this paper, we assume X is the set of legitimate images (and not garbage images or ambiguous images).

Given a label y, let $Cat(y)$ denote the categorical probability distribution with the component for y set to 1 and all else 0.

Robustness: We introduce some concepts from PAC learning [1], in order to present the formal results in this section. It is assumed that data points arise from a fixed but unknown distribution $\mathcal{P}$ over X. We denote the probability mass over a set $Z \subset X$ as $\mathcal{P}(Z)$. A loss function $l(y_x, C(x))$ captures the loss of predicting $C(x)$ when the true label for x is y_x. As we are focused on classification, we restrict ourselves to the ideal 0/1 loss, that is, 1 for mis-classification and 0 otherwise. A classifier C is chosen that minimizes the empirical loss over the n training data points $\sum_{i=1}^{n} l(y_{x_i}, x_i)$. Given enough data, PAC learning theory guarantees that C also minimizes the expected loss $\int_X l(y_x, C(x))\mathcal{P}(x)$. Given, 0/1 loss this quantity is just $\mathcal{P}(M_C(X))$, where $M_C(X) \subset X$ denote the region where the classifier C mis-classifies. Accuracy is then just $1 - \mathcal{P}(M_C(X))$. In this paper we assume that the amount of data is always enough to obtain low expected loss. Observe that a classifier can achieve high accuracy (low expected loss) even though its predictions in the low probability regions may be wrong [21]. All classifier families have a capacity that limits the complexity of separators that they can model; the capacity value is known only for simple classifiers [1]. Previous work [9] has conjectured that adversarial examples abound due to the low capacity of the classifier family used. See Fig. 2A.

Adversarial exploitable space: Define the adversarial exploitable space:

$$E_{C,\epsilon}(X) = M_C(X) \cup \{x \mid \overline{sim}(x, M_C(X)) \leq \epsilon\} \ ,$$

where $\overline{sim}$ is a *dissimilarity* measure that depends on the domain and $\overline{sim}(x, M_C(X))$ denotes the lowest dissimilarity of x with any data point in $M_C(X)$. For image classification $\overline{sim}$ can just be the l_2 (Euclidean) distance: $\sqrt{\sum_i (x_i - x'_i)^2}$ where i indexes the pixels. $E_{C,\epsilon}(X)$ includes all points that are either mis-classified or can be mis-classified by a minor ϵ-perturbation. Observe that we posit that any already present mis-classifications of the classifier is exploitable by the adversary, e.g., if a stop sign image in a dataset is mis-classified then an adversary can simply use this image as is to fool a classifier.

Robustness definition: Robustness is simply defined as $1 - \mathcal{P}(E_{C,\epsilon}(X))$. First, observe that robustness is a strictly stronger concept than accuracy, i.e., accuracy is always higher than robustness. We believe this property makes our definition more natural than other current definitions. Another readily inferable property is that a classifier C' with $M_{C'}(X) \subset M_C(X)$ has higher robustness than C in the same stochastic setting. We use this property later to justify our defense. Next, we elaborate on a number of subtle aspects of the definition.

First, a 100% robust classifier can still have $M_{C'}(X) \neq \phi$. This is because robustness is still defined w.r.t. the data distribution $\mathcal{P}$. For example, large compact regions R of zero probability with small sub-region of erroneous prediction far away from the boundary of R can still make robustness 100%. On the other hand, $M_{C'}(X) = \phi$ provides 100% robustness for any $\mathcal{P}$. Second, as shown in Fig. 2, low capacity classifiers cannot model complex separators, thus, large capacity is required to achieve robustness. A 100% robust classifier is practically

impossible due to large data requirement of high capacity classifier family. On the other hand, large capacity but limited data causes over-fitting [1]. Thus, there is a delicate balance between capacity and amount of data, which is not well understood for DNNs. Third, robustness may appear to be computable by calculating the accuracy for the test set and for the adversarially perturbed test set, as done in all prior work. However, this relies on the assumption that the attack discovers *all* perturb-able points. An analysis of computing robustness is beyond the scope of this paper.

Lastly, compared to past work [24,8], our robustness has a clear relation to accuracy and not orthogonal to it. Also, we use the ideal 0/1 loss function rather than an approximate loss function l (often used in training due to smoothness) as used in other definitions [17,7,12]. We posit that the 0/1 loss measures robustness more precisely, as these other approaches specify the adversary goal as perturbing in order to produce the maximum loss within an ϵ ball $B(x, \epsilon)$ of any given point x, with the defender expected loss defined as $\int_X \max_{z \in B(x,\epsilon)} l(y_x, C(z))\mathcal{P}(x)$, where l is the loss function used to train the classifier. For ease of optimization, typically, l is a smooth function approximation of the 0/1 loss. However, this means that even if the class is same throughout the ϵ ball, with a varying l the adversary still conducts a "supposed" attack and increases loss for the defender without flipping labels. For example, the well-known hinge loss varies rapidly within one of the classes and could overestimate the loss for defender and hence underestimate robustness.

Robustness vs Accuracy: Finally, we analyze the relation between accuracy and robustness. First, it is straightforward to check from definition that $1 - a$ robustness implies $1 - a$ accuracy. However, the converse is not true, and the example in Figure 1 is a proof that the converse does not hold. In this example, assume the data is distributed uniformly over the 2d space and the true separator happens to be close to a large fraction of points in the given 2d space (an extreme example is where the separator is within ϵ of every point in the underlying space, in which case the separator is an ϵ-net). Then, the small misclassified parts (hence high accuracy) near the learned separator expands to produce a large adversarial exploitable space (low robustness). Some defenses in literature [12] have tried to guarantee that the learned classifier does not change labels within an ϵ ball $B(x, \epsilon)$ of any (or most) given training data point x. This example (particularly, in the extreme case where the separator is an ϵ-net) shows that it may be the case the label changes legitimately in an ϵ ball around any data point x. Thus, the nature of the underlying ground truth is an important

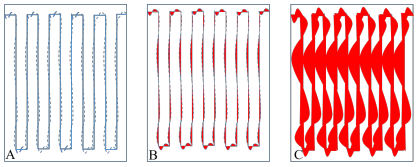


Fig. 1. Robustness vs Accuracy. (A) shows a piecewise linear classifier (solid line) is not able to exactly match the non-linear boundary (dashed line). (B) shows that the mis-classification space (red/shaded area) is small, hence accuracy is high. (C) shows that adversarial exploitable space (red/shaded area) is large, hence robustness is low.

factor for robustness and its relation with accuracy. In the analysis in the next paragraph, we show that the data distribution is also another important factor.

Next, we analyze if accuracy is ever suitable to capture robustness. First, we make a few mild technical assumptions that there exists a density $p(x)$ for the data distribution $\mathcal{P}$ over X, X is a metric space with metric d and $vol(X) = 1$.

Theorem 1. *$1 - a$ accuracy implies at least $1 - a - (\nu + K\epsilon/T)\mathcal{P}(\underline{E_{C,\epsilon}}(X) \backslash M_C(X))$ robustness for any output C if (1) For all $x \in X$, $\overline{sim}(x, x') \geq Td(x, x')$ for some $T > 0$, (2) $M_C(X)$ lies in a low density region, that is, for all $x \in M_C(X)$ we have $p(x) \leq \nu$ for some small ν, and (3) $p(x)$ is K-Lipschitz, that is, $|p(x) - p(x')| \leq Kd(x, x')$ for all $x, x' \in X$.*

The first two conditions above are quite natural. In simple words, these two conditions state that dissimilarity increases with distance (high T) and the regions where the classifier predicts badly has low amount of data in the data-set (low ν). However, the final condition may not be satisfied in many natural settings. This condition states that the data distribution must not change abruptly (low K). This is required as the natural behavior of most classifiers is to predict bad in a low data density region and if this region is near a high data density region, the adversary can successfully modify the data points in the high density region causing loss of robustness. But in high dimensional spaces, data distribution is quite likely to be not distributed smoothly with many pockets or sub-spaces of zero density as pointed out in a recent experimental work [22]. Thus, data distribution, especially in the region around the mis-classified space, has a huge effect on robustness.

Intuition Behind Attacks: *Any* adversarial example generation A can be seen as a distribution transformer F_A such that acting on the data distribution $\mathcal{P}$ the resultant distribution $F_A(\mathcal{P})$ has support mostly limited to $M_C(X)$. The support may not completely limited to $M_C(X)$ as the attacks are never 100% effective. Also, attacks typically aim to find points in $M_C(X)$ that are close to given images in the original dataset. See Fig. 2B for an illustration. As an example, a recent work [2] provides the adversarial transformation network (ATN) technique, which trains a DNN to produce adversarial examples. ATN is essentially a neural network representation of a distribution transformer function F. For other attack techniques like FGSM and CW, the function F is evaluated for each sample (data point) by solving an optimization problem, utilizing gradients of the classifier in case of FGSM.

High vs Low Perturbation: Lastly, we show in our experiments that FGSM produces adversarial examples whose perturbations are at least an order of magnitude higher than CW. We categorize FGSM as a high perturbation attack. On the other hand, the attack CW produces adversarial perturbation with very small perturbations; we call such attacks low perturbation attacks.

DLN: Our first defense approach is to insert a neural network DLN D between the input and classifier so that D sanitizes the input enabling the classifier to correctly classify the input. Each data point for training DLN has three parts: x' is the image to sanitize (input), x is the expected output and y_x is the correct

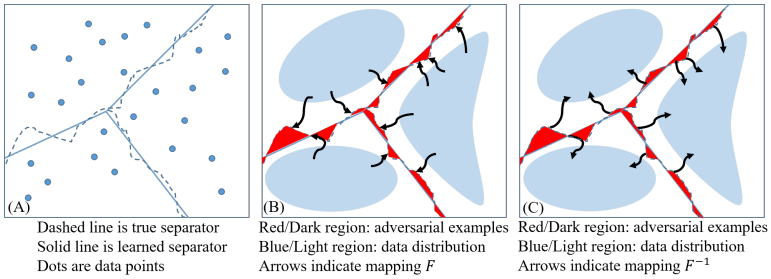


Fig. 2. Intuition behind attacks and DLN. (A) shows a linear classifier (low capacity) is not able to accurately model a non-linear boundary. (B) shows attacks as the distribution mapping function F. (C) shows that DLN does the reverse mapping of attacks.

label of x. The x's are always images from the provided dataset, and there are two possibilities for x': (1) $x' = x$ so that DLN attempts to satisfy $C(D(x)) = y_x$, even if $C(x) \neq y_x$, and (2) $x' = A(x)$ so that DLN undoes the attack and make the classifier C correctly classify $D(x')$. Thus, for training DLN, the original training set is attacked to produces $A(x)$'s and the training set for DLN is twice the original training set, with one half having $x' = x$ and another half having $x' = A(x)$. We formulate a loss function for DLN that has two terms: $\overline{sim}(x, D(x'))$ aims to produce output $D(x')$ close to x and $H(Cat(y_x), C_p(D(x')))$ aims to make the classifier output on $D(x')$ be the same as y_x. The loss function is

$$\overline{sim}(x, D(x')) + H(Cat(y_x), C_p(D(x'))) \ .$$

Note that the attack A is used as a black box here to generate training data and is not a part of the loss function. After training the DLN, our new classifier is C^1 which is C prepended by the DLN D, represented as $C^1 = \boxed{D \rightarrow C}$. The working of DLN can be interpreted as an inverse map F^{-1} for the mapping F induced by the attack A. See Figure 2C for an illustration. For the image classification problem we use the l_2 distance for $\overline{sim}$.

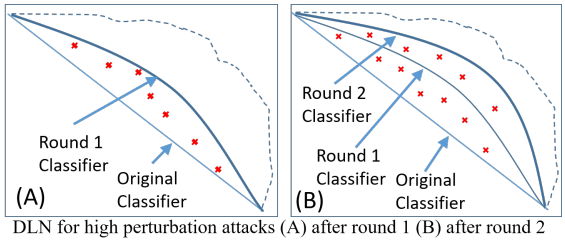
Intuitively, as C^1 correctly classifies the adversarial examples in addition to correctly classifying more data points than C, it shrinks the mis-classification space of C^1 to within the mis-classifications of C. As argued when defining robustness, this leads to an increase in robustness for C^1 over C. See Fig. 3A for an illustration of how the mis-classification space of C^1 shrinks over that of C. Fig. 3(A and B) also motivate the repeated DLN below.

Repeated DLN: While the robustness of C^1 could be higher than C, unless the attack used discovers all potential exploitable data points of C, there may still be a lot of exploitable data points for C^1. This is why attacks on C^1 are still effective (see experiments). One approach to overcome this is to repeatedly attack and discover more of the exploitable space. DLN allows for efficient *modular retraining* (without retraining the classifier) of the DLN in rounds as follows: starting from $i = 0$, (A) attack the classifier $C^i = \boxed{D^{i-1} \rightarrow C}$ ($C^0 = C$) at round

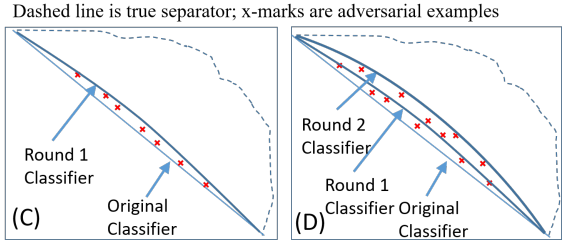
i generating adversarial training data T_A from the original training data T; (B) (re)train the DLN with data T_i to get D^i, where T_i is formed by augmenting all the past data T_{i-1} ($T_0 = \phi$) with T_A and T. Then, repeat step A with $i = i + 1$.

Observe that we add copies of original training data at each step i in order to prevent the adversarial data from swamping out the original training data. See Fig. 3 for an illustration of how repeated DLN works. Intuitively, in each round the exploitable space reduces providing less space for the attack to be successful. This makes the high perturbation attacks less effective within a few rounds.

In this attack-defense competition, in every round the dataset used to train the DLN grows. Practically, this requires DLN to have a large capacity in order to be effective; also depending on the capacity and the size of dataset over or under fitting problems could arise, which needs to be taken care of in practice. Also, the training becomes more expensive over rounds with increasing data size. In particular, low perturbation attacks are not defeated within few rounds. We observed very small improvements with the low perturbation CW attack over rounds, as illustrated visually in Fig. 3 (C and D). The main reason, as shown in Fig. 3, is that low perturbation attacks only expose a very small volume of mis-classified space, thus, fixing only a small part of the mis-classified space in every round of repeated DLN. Further, low perturbation attacks only need a small volume of mis-classfied space near the classifier boundary to be successful. It would require a huge number of rounds for repeated DLN to reduce the mis-classified space to such a small volume that cannot be attacked. This motivates our next approach of noise augmented classifier.



DLN for high perturbation attacks (A) after round 1 (B) after round 2
Dashed line is true separator; x-marks are adversarial examples



DLN for low perturbation attacks (C) after round 1 (D) after round 2

Fig. 3. Intuition behind working of repeated DLN against high and low perturbation attacks. (A),(B) shows a high perturbation attack causes a faster improvement in resultant classifier. Beyond some rounds the attack does not work as it can only find adversarial examples with high perturbation. (C),(D) shows a low perturbation attack causes a slow improvement in resultant classifier.

NAC: Fig. 3 provides a hint on how to overcome low perturbation attacks. Such attacks rely a lot on the knowledge about the exact classifier boundary in order to add a very small perturbation and yet change the label of input image. Thus, randomly shifting the classifier boundary may help against low perturbation attacks. We shift the boundary randomly by adding a small noise to the logits of

the DNN (the last layer of the DNN that yields the class probabilities) C at *inference time only* calling the resultant classifier a noise augmented classifier (NAC) C^N. Through experimentation we chose a Gaussian noise with mean 0 and standard deviation 1. This noise is small enough that it does not affect the classification of original data points by much, but is able to mis-lead the low perturbation attack. Also, following our explanation, NAC should not provide any defense against high perturbation attacks, which we observe in our experiments.

As NAC defense is a inference time (runtime) technique, hence, NAC can be used in conjunction with any other training time defense, such as DLN. Further, according to our attack model, the adversary does not have access to the noise added to the logit layer. However, a natural idea to bypass the NAC defense is to take the average of multiple logit outputs for the same input image (to cancel the randomness) and then use the average logits for the CW attack. We show experimentally that this improved attack does not work effectively.

3 Experiments

All our experiments were conducted using the Keras framework on a NVIDIA K40 GPU. We consider two classifiers one for MNIST and one for CIFAR-10: we call them C_M and C_C. These classifiers are variants of well-known architectures. We show two attacks: FGSM and CW to show their categorization into high and low perturbation attacks. Attacks were used with default parameters. CW, while slow to run, has been referred to in the literature [25] as the best attack till date, while FGSM runs extremely fast. Observe that all these attacks work against a given classifier C, thus, we use the notation $A(C, .)$ to denote the attack A acting on an image x to produce the adversarial example $A(C, x)$ (A can be any of the three attacks). $A(C, Z)$ denotes the set of adversarial examples $\{A(C, x) \mid x \in Z\}$. We report accuracies on various test sets: (A) original test dataset (OTD): this is the original test set from the dataset under consideration, (B) $A(C, OTD)$ is the adversarially perturbed test data using attack A, for example, this could be FGSM(C_M, OTD). We also report distortion numbers as done in prior work [5]. Distortion is the average over all test images of the l_2 distance between the original and perturbed image.

Test data type	Accuracy	Distortion
FGSM(C_M, OTD)	0.72 %	14.99
CW(C_M, OTD)	0.03 %	1.51
FGSM(C_C, OTD)	4.21 %	10.03
CW(C_C, OTD)	0 %	0.18

Table 1. Attacks on MNIST and CIFAR-10

Table 1 shows the result of attacks using FGSM and CW on MNIST and CIFAR-10. It can be seen that FGSM produces higher distortion than CW. For defense, we denote the new classifier using the DLN with i rounds of training as C_M^i (for MNIST) or C_C^i (for CIFAR); analogously for NAC we get C_M^N or C_C^N, and for both DLN and NAC we get $C_M^{i,N}$ or $C_C^{i,N}$. Also, we test accuracies on

the adversarially perturbed test data against the newer classifiers, e.g., following our convention one such dataset would be denoted as $A(C_M^i, OTD)$.

DLN Defense Against Single Attack: Table 2 shows the results when DLN is trained to defend against FGSM or CW using MNIST dataset to yield a new classifier C_M^1. As expected, the accuracy on OTD drops slightly for all the cases. Moreover, when attacked again the new classifier C_M^1 is not resilient to attacks as shown by the low accuracies for FGSM(C_M^1, OTD) and CW(C_M^1, OTD). One number that stands out is the success of the new classifier C_M^1 in correctly classifying the adversarial examples CW(C_M, OTD) generated by CW (row 5). This supports our hypothesis that CW is sensitive to the exact classifier boundary, and a newer classifier C_M^1 with a slightly different boundary is able to correctly classify prior adversarial examples. Of course, CW is able to attack C_M^1 again successfully. For FGSM, we show next that the performance of the classifier greatly improves when DLN is repeatedly trained against FGSM.

DLN Trained	Test data type	Accuracy	Distortion
FGSM	OTD	96.77 %	–
FGSM	FGSM(C_M, OTD)	88.5 %	4.55
FGSM	FGSM(C_M^1, OTD)	13.75 %	6.98
CW	OTD	98.6 %	–
CW	CW(C_M, OTD)	95.42 %	5.77
CW	CW(C_M^1, OTD)	0.14 %	3.5

Table 2. Performance of DLN prepended C_M^1 for MNIST

Round	Acc. OTD	Acc. FGSM(C_M^i, OTD)	Distortion
0	99.36 %	0.72 %	14.99
1	97.70 %	13.70 %	13.63
2	97.61 %	24.86 %	14.58
3	97.95 %	43.39 %	14.73
4	97.79 %	52.88 %	14.57
5	97.77 %	56.57 %	14.52

Table 3. DLN trained repeatedly against FGSM for MNIST

Repeated DLN: Next, we run DLN repeatedly as described earlier. We cut off the experiments when a single round took more than 48 hours to solve. We show the results for MNIST in Table 3 showing a clearly increasing trend in accuracy on adversarial examples

Attack	Test data type	Accuracy	Distortion
-	OTD(MNIST)	99.36 %	-
CW	CW(C_M^N, OTD)	93.60 %	1.49
FGSM	FGSM(C_M^N, OTD)	0.74 %	14.99
-	OTD(CIFAR)	84.67 %	-
CW	CW(C_C^N, OTD)	77.70 %	0.17
FGSM	FGSM(C_C^N, OTD)	4.19 %	10.04

Table 4. Accuracy of NAC for MNIST and CIFAR-10

produced by FGSM attacking the newer classifier, revealing increasing robustness. For CIFAR, the approach becomes too computationally expensive within two rounds. Further, as stated earlier, DLN does not show much improvement against low perturbation attacks like CW. We tackle that next using the NAC defense.

NAC Defense: The NAC defense produces a new classifier C_M^N for MNIST and C_C^N for CIFAR. The second and fifth row in Table 4 shows that the NAC defense leads to a failure of the CW attack. Further, the new classifier's accuracy on the original test data-set is nearly unchanged. However, it can also be observed that the new classifier is not resilient to attack by FGSM, as shown by the third and sixth row, which follows from the intuition in Fig. 3.

n	Adv. accuracy	Distortion
500	95.14 %	1.51
5000	82.07 %	1.51

Table 5. NAC classifier C_M^N against improved CW for MNIST

As stated earlier, a natural idea to attack NAC would be to query an image n times and then average the logits before using it for the CW attack. This improved attack does make CW more effective but not by much. Table 5 shows that the accuracy on the adversarial example generated for C_M^N remains high. Moreover, more queries make it more difficult to conduct attacks in practice (e.g., query limited adversary), while also causing an increase (2% with 5000 samples) in the already high runtime of CW.

Defense Against Multiple Attacks: Finally, we show that DLN and NAC can work together. We show the accuracy on the adversarial example generated in each round of DLN repetition when the classifier C^i after each round is augmented with NAC $C^{i,N}$ and then attacked by FGSM and CW both. See Table 6. One observation is that NAC's performance decreases slightly over rounds stabilizing at 79%, while the accuracy for original test set and FGSM perturbed test set stays almost exactly same as Table 3.

Round	Acc. OTD	Acc. FGSM $(C_M^{i,N}, OTD)$	Acc. CW $(C_C^{i,N}, OTD)$
0	99.36 %	0.72 %	94.2
1	97.70 %	13.72 %	93.7
2	97.73 %	24.28 %	84.7
3	97.60 %	43.20 %	83.3
4	97.64 %	53.17 %	79
5	97.73 %	56.45 %	79.3

Table 6. Classifier trained repeatedly against FGSM for MNIST and augmented with NAC in each round

4 Related Work and Summary

A thorough survey of security issues in machine learning, including types of attacks, is present in surveys [23,20,3] and some of the first results appeared in [16]. Here we discuss only the most closely related defense work.

Defense: Defense techniques can be roughly categorized into training time techniques that do (1) adversarial (re)training, which is adding back adversarial examples to the training data and retraining the classifier, often repeatedly [14], or modifying loss function to account for attacks [11,12]; (2) gradient masking, which targets that gradient based attacks by trying to make the gradient less informative [19]; (3) input modification, which are techniques that modify (typically lower the dimension) the feature space of the input data to make crafting adversarial examples difficult [25]; (4) game-theoretic formulation, which modifies the loss minimization to a constrained optimization with constraints provided by adversarial utility in performing perturbations [13], and (5) filtering and de-noising [15,10,6,18], which aims to filter or de-noise adversarial examples.

Our DLN approach differs from the first four kinds of defense as our approach never modifies the classifier or its inputs but adds a sanitizer (DLN) before the classifier. Our approach, while similar in spirit to adversarial re-training, increases the capacity of the resultant classifier C^i, so that it can model more complex separators which is not achieved when the classifier family stays the same. Also, the re-training is not of the whole network but just the DLN module, which is faster than re-training large classifiers. Next, unlike the fifth kind of defense, our goal for DLN is targeted sanitization and not generic de-noising; we aim to reduce mis-classifications which means correctly classifiying adversarial examples as well as original mis-classifications. More significantly, attempts such as MagNet [18] reach a wrong conclusion that they defend against CW [4]. In contrast, we repeatedly attacked the new DLN classifier showing that a sanitizing approach like DLN cannot defend against low perturbation attacks.

As far as we know, NAC being a runtime technique, is novel and entirely different from training time approaches; moreover, NAC is compatible with any other training time approach. Interestingly, the DLN and NAC approaches can be used with any classifier that outputs class probabilities and not just DNNs.

Summary We provided a new perspective of the adversarial examples defense problem with a formal intuition of how our approach works, using which we were able to defend simultaneously against multiple attacks including the potent CW attack. We identified two classes of attacks: high and low perturbation, and proposed the DLN technique to defend against high perturbation attacks and NAC to defender against low perturbation attacks. Extensions of our theory and tuning of the application framework provides rich content for future work.

References

1. Anthony, M., Bartlett, P.L.: Neural Network Learning: Theoretical Foundations. Cambridge University Press, New York, NY, USA, 1st edn. (2009)
2. Baluja, S., Fischer, I.: Adversarial transformation networks: Learning to generate adversarial examples. CoRR **abs/1703.09387** (2017), http://arxiv.org/abs/1703.09387
3. Biggio, B., Roli, F.: Wild patterns: Ten years after the rise of adversarial machine learning. arXiv preprint arXiv:1712.03141 (2017)

4. Carlini, N., Wagner, D.: Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples. arXiv preprint arXiv:1711.08478 (2017)
5. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: Security and Privacy (SP), 2017 IEEE Symposium on. pp. 39–57. IEEE (2017)
6. Chen, X., Li, B., Vorobeychik, Y.: Evaluation of defensive methods for DNNs against multiple adversarial evasion models (2016), https://openreview.net/forum?id=ByToKu9ll&noteId=ByToKu9ll
7. Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., Usunier, N.: Parseval networks: Improving robustness to adversarial examples. arXiv preprint arXiv:1704.08847 (2017)
8. Fawzi, A., Fawzi, O., Frossard, P.: Fundamental limits on adversarial robustness. In: Proc. ICML, Workshop on Deep Learning (2015)
9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. CoRR **abs/1412.6572** (2014), http://arxiv.org/abs/1412.6572
10. Grosse, K., Manoharan, P., Papernot, N., Backes, M., McDaniel, P.: On the (statistical) detection of adversarial examples. arXiv preprint arXiv:1702.06280 (2017)
11. Huang, R., Xu, B., Schuurmans, D., Szepesvári, C.: Learning with a strong adversary. arXiv preprint arXiv:1511.03034 (2015)
12. Kolter, J.Z., Wong, E.: Provable defenses against adversarial examples via the convex outer adversarial polytope. arXiv preprint arXiv:1711.00851 (2017)
13. Li, B., Vorobeychik, Y.: Feature cross-substitution in adversarial classification. In: Advances in neural information processing systems. pp. 2087–2095 (2014)
14. Li, B., Vorobeychik, Y., Chen, X.: A general retraining framework for scalable adversarial classification. arXiv preprint arXiv:1604.02606 (2016)
15. Li, X., Li, F.: Adversarial examples detection in deep networks with convolutional filter statistics. arXiv preprint arXiv:1612.07767 (2016)
16. Lowd, D., Meek, C.: Adversarial learning. In: ACM SIGKDD. ACM (2005)
17. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 (2017)
18. Meng, D., Chen, H.: Magnet: a two-pronged defense against adversarial examples. In: ACM Conference on Computer and Communications Security (2017)
19. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against deep learning systems using adversarial examples. arXiv preprint arXiv:1602.02697 (2016)
20. Papernot, N., McDaniel, P., Sinha, A., Wellman, M.: Towards the science of security and privacy in machine learning. arXiv preprint arXiv:1611.03814 (2016)
21. Sinha, A., Kar, D., Tambe, M.: Learning adversary behavior in security games: A PAC model perspective. In: Conference on Autonomous Agents & Multiagent Systems (2016)
22. Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., McDaniel, P.: The space of transferable adversarial examples. arXiv preprint arXiv:1704.03453 (2017)
23. Tygar, J.: Adversarial machine learning. IEEE Internet Computing **15**(5), 4–6 (2011)
24. Wang, B., Gao, J., Qi, Y.: A theoretical framework for robustness of (deep) classifiers under adversarial noise. arXiv preprint arXiv:1612.00334 (2016)
25. Xu, W., Evans, D., Qi, Y.: Feature squeezing: Detecting adversarial examples in deep neural networks. arXiv preprint arXiv:1704.01155 (2017)